

LISTING OF THE CLAIMS

Please change the status of the claims of the present application as set forth below. This listing of claims will replace all prior versions and listings of claims in the application.

By way of overview claims 1-42 are currently pending. The status of the pending claims is indicated below:

- a) Claims 18-27 and 35-42 are original; and
- b) Claims 1-17 and 28-34 are withdrawn.

Listing of Claims

1. (Withdrawn) A method, comprising:
receiving an object, the object being structured according to a first protocol;
evaluating at least a portion of the object against one or more queries structured according to a second protocol, the second protocol being different than the first protocol;
and
wherein the evaluating step is performed without serializing the object to reconstruct the object according to the second protocol.

2. (Withdrawn) The method as recited in claim 1, wherein:
the first protocol further comprises Common Language Runtime (CLR) protocol;
and
the second protocol further comprises eXtensible Markup Language (XML) protocol.

1 3. (Withdrawn) The method as recited in claim 1, wherein the evaluating step
2 further comprises:

- 3 accessing an infoset that maps to the object;
4 locating a value being evaluated by the query using the infoset; and
5 evaluating the query according to at least the located value.
6

7 4. (Withdrawn) The method as recited in claim 3, wherein the accessing the
8 infoset step further comprises retrieving the infoset from memory.
9

10 5. (Withdrawn) The method as recited in claim 3, further comprising constructing
11 at least a portion of the infoset.
12

13 6. (Withdrawn) The method as recited in claim 3, wherein the accessing the
14 infoset step further comprises retrieving a portion of the infoset from memory, and the
15 method further comprises constructing at least an additional portion of the infoset.
16

17 7. (Withdrawn) The method as recited in claim 3, further comprising storing the
18 infoset for re-use when the object is encountered in another input.
19

20 8. (Withdrawn) The method as recited in claim 1, wherein the evaluating step
21 further comprises:

- 22 creating a template according to the second protocol;
23 locating a template element that corresponds with a query value;
24 locating an object property associated with the template element; and
25 evaluating a value of the object property against the query value.

1
2 9. (Withdrawn) The method as recited in claim 8, further comprising the step of
3 generating one or more opcodes that initiate the evaluating step when the object is
4 encountered in a subsequent query evaluation process.

5
6 10. (Withdrawn) The method as recited in claim 1, further comprising:
7 generating one or more opcodes that, when executed, perform at least a portion of
8 the evaluating step;
9 compiling and executing the one or more opcodes at runtime; and
10 wherein executing the one or more compiled opcodes retrieves property values
11 and fields directly from the object for evaluation.

12
13 11. (Withdrawn) A system, comprising:
14 one or more filters, each filter being structure according to a first protocol;
15 an input module configured to receive an input for evaluation against at least a
16 query, the input including an object that is structured according to a second protocol that
17 is different from the first protocol;
18 a mapping module configured to map one or more properties of the object to a
19 template arranged according to the first protocol without serializing the object or any
20 portion thereof; and
21 a filter engine configured to locate a query value in the object by referring to the
22 template to locate an object value corresponding to the query value, to evaluate the query
23 value against the object value, and to return a result of the evaluation.

12. (Withdrawn) The system as recited in claim 11, wherein the mapping module is further configured to utilize an infoset model corresponding to the object to map the object properties to the template.

13. (Withdrawn) The system as recited in claim 12, wherein the mapping model is further configured to retrieve at least a portion of the infoset model from memory.

14. (Withdrawn) The system as recited in claim 12, wherein the mapping module is further configured to construct at least a portion of the infoset.

15. (Withdrawn) The system as recited in claim 11, further comprising an opcode generation module configured to generate one or more opcodes that, when executed, perform one or more steps taken in the process of evaluating the object so that the one or more opcodes can be used in subsequent evaluations regarding the object to automatically perform the one or more steps.

16. (Withdrawn) The system as recited in claim 15, wherein the opcodes are compiled and executed at runtime.

17. (Withdrawn) The system as recited in claim 11, wherein the query is an eXtensible Markup Language (XML) query and the object is a Common Runtime Language (CLR) object.

18. (Original) One or more computer-readable media comprising computer-executable instructions that, when executed on a computer, perform the following steps:

1 creating an object model that maps object properties to an object template that
2 conforms to a query protocol, wherein the object conforms to an object protocol that is
3 different than the query protocol;
4 identifying a query value;
5 referencing the object template to locate an object property corresponding to the
6 query value;
7 identifying a property value assigned to the object property; and
8 evaluating the property value against the query value to determine if the query is
9 satisfied by the property value.

10
11 19. (Original) The one or more computer-readable media as recited in claim 18,
12 wherein the object properties are mapped to the object template and the property value is
13 evaluated against the query without serializing object data.

14
15 20. (Original) The one or more computer-readable media as recited in claim 18,
16 wherein the query protocol is eXtensible Markup Language (XML) and the object
17 protocol is Common Language Runtime (CLR).

18
19 21. (Original) The one or more computer-readable media as recited in claim 18,
20 wherein the object model is an infoset model that corresponds to the object.

21
22 22. (Original) The one or more computer-readable media as recited in claim 18,
23 wherein the creating an object model further comprises creating only a portion of the
24 object model that is necessary to discover the object property tested by the query.
25

23. (Original) The one or more computer-readable media as recited in claim 18, further comprising storing the object model so that it can be retrieved for future query evaluations against the object to avoid having to re-create the object model.

24. (Original) The one or more computer-readable media as recited in claim 18, wherein the creating an object model further comprises retrieving a partially completed version of an object model and augmenting the object model to an extent necessary to locate the object property that corresponds to the query value.

25. (Original) The one or more computer-readable media as recited in claim 18, further comprising:

generating one or more opcodes to perform the referencing step, the identifying a property value step and the evaluating step;

storing the one or more opcodes in memory; and

wherein the opcodes can be retrieved and utilized to perform similar steps in a subsequent query evaluation involving the object.

26. (Original) The one or more computer-readable media as recited in claim 25, wherein the opcodes can be compiled and executed dynamically at runtime.

27. (Original) The one or more computer-readable media as recited in claim 18, wherein the query further comprises an XPath filter.

28. (Withdrawn) A filter engine stored on one or more computer-readable media, comprising:

one or more filters derived from a query language, each filter containing at least one condition specifying a query value that can be compared to one or more filter engine inputs;

an input function configured to receive an input for comparison against one or more of the filters wherein the input may contain an object derived from an object language that is different from the query language;

a mapping module configured to reference an object info set and map at least one object property identified in the object info set to an object template that is in accordance with the query language without serializing data included in the object; and

wherein the filter engine is further configured to locate an object property value by referencing the object template to locate the object property in the object info set, and to evaluate the object property value against the query value.

29. (Withdrawn) The filter engine as recited in claim 28, wherein the mapping module is further configured to create the object info set or to augment a partial object info set retrieved from memory.

30. (Withdrawn) The filter engine as recited in claim 28, wherein the mapping module is further configured to store the object info set if the mapping module has changed the object info set.

31. (Withdrawn) The filter engine as recited in claim 28, further comprising:
an opcode generation module configured to generate one or more opcodes to automatically perform one or more of the filter engine procedures upon a first encounter with the object;

1 an opcode store for storing opcodes generated by the opcode generation module;
2 and

3 wherein the one or more opcodes can be used to perform the one or more filter
4 engine procedures upon a subsequent encounter with the object.

5
6 32. (Withdrawn) The filter engine as recited in claim 31, wherein the opcodes can
7 be dynamically compiled and executed at runtime.

8
9 33. (Withdrawn) The filter engine as recited in claim 28, wherein the query
10 language is XPath and the object language is a Common Language Runtime (CLR)
11 language.

12
13 34. (Withdrawn) The filter engine as recited in claim 28, wherein the query
14 language and the object language are hierarchically structured so that hierarchy levels of
15 the object info set have corresponding level in the hierarchy of the object template.

16
17 35. (Original) A method, comprising:
18 mapping object properties to template elements;
19 identifying a query value in a query against which the object is to be evaluated;
20 referencing the template to identify an element corresponding to the query value;
21 identifying an object property value corresponding to the identified template
22 element;

23 comparing the object property value to the query value to evaluate at least a
24 portion of the query; and
25

1 wherein the object is derived from an object language, the query is derived from a
2 query language, and the steps are accomplished without serializing data included with or
3 referenced by the object.

4
5 36. (Original) The method as recited in claim 35, wherein the object language
6 further comprises a Common Language Runtime (CLR) language.

7
8 37. (Original) The method as recited in claim 35, wherein the query language
9 further comprises eXtensible Markup Language (XML).

10
11 38. (Original) The method as recited in claim 35, wherein the query language is
12 XPath.

13
14 39. (Original) The method as recited in claim 35, further comprising using an
15 object infoset model that references object properties to map the object properties to
16 template elements.

17
18 40. (Original) The method as recited in claim 39, further comprising building the
19 infoset model at least to the extent necessary to identify the correct object property value.

20
21 41. (Original) The method as recited in claim 39, further comprising retrieving the
22 infoset model from memory.

23
24 42. (Original) The method as recited in claim 39, further comprising:
25 retrieving a partial infoset model from memory; and

1 if the object property value cannot be identified from the partial infoSet,
2 augmenting the infoSet model at least to the extent necessary to identify the correct object
3 property value.
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25